

The principle of cognitive action

Preliminary experimental analysis

Marco Gori, Marco Maggini, Alessandro Rossi

Department of Information Engineering and Mathematical Sciences,

University of Siena, Italy

May 20, 2015

Abstract

In this document we shows a first implementation and some preliminary results of a new theory, facing Machine Learning problems in the frameworks of Classical Mechanics and Variational Calculus. We give a general formulation of the problem and then we studies basic behaviors of the model on simple practical implementations.

Contents

1	Introduction	3
2	Formulation of the problem	3
3	Construction of Cost Functional	4
4	First Application	5
4.1	First Order Operator	5
4.1.1	Experimental results	6
4.1.2	Choice of solutions	19
4.2	Second Order Operator	20
4.2.1	Experimental results	21
4.2.2	Choice of solutions	32
4.3	Sampling-step τ , Parameter θ and number of Impulses	33
5	First application on ANNs	37
5.1	One dimension functions	38
5.2	Two dimensions functions	38
5.3	Vowels Classifications	39
6	Conclusions	40
7	Appendix	41
7.1	General solution and coefficients	41
7.2	From solution to parameters	43
7.2.1	First Order	43
7.2.2	Second Order	44
7.3	From continuos to discrete model	45

1 Introduction

Many real world phenomena could be interpreted in an on-line scenario within Machine Learning theory. In long-life learning problems, various approaches have been developed to deal with the large amounts of data and the exploitation of their time correlation. Usually, some difficulties arise in the storage of data and in going after the intrinsic information coming from data during time. These aspects could suggest a more natural approach to learning, in a joint theory among Mechanics, Variational Calculus and Statistics. However, we postpone a deeply analysis of these ideas at a theoretical level, focusing on a first practical implementation, to create a connection between this new ideas and some applications on existing structures, which could be useful in these preliminary steps.

We briefly introduce basics ideas of this theory, first formulated in [1]. The concept of dissipation is well formulated in [2], whereas an summing up of this report and an experimental analysis on standard benchmark can be find in [3]. A further theoretical abstraction applied to similar environment is proposed in [4]. In this document, we will give a slightly theoretical formulation in order to allow us to go straight to the practical implementation issues. We study the meaning of the model parameters on artificial problems, using a linear function, and then we try some experiments with simple Neural Networks.

An important result is the fact that we can choose arbitrarily the memory of our system by a parameter, avoiding the hardware memory storage problems. Indeed, the trend of the system is influenced by the information coming from each example during an adjustable interval of time.

2 Formulation of the problem

We study the case in which we want to learn a function f that aims to represent the behavior of a features representation \mathbf{u} of the spatio-temporal domains \mathcal{D} . If we assume the temporal domain $\mathcal{T} = [0, \infty)$ and $\mathcal{X} \subseteq \mathbb{R}^D$ then we have $\mathcal{D} = \mathcal{T} \times \mathcal{X}$ and $\mathbf{u} : \mathcal{D} \rightarrow \mathbb{R}^d$ so that f has input \mathbf{u} and depends on a set of weights \mathbf{W} , like for example if f is described by an Artificial Neural Network. The problem consists on learning the parameters \mathbf{W} under some assumption, i.e. by minimizing a cost functional \mathcal{L} composed by a penalty term and a regularization one. Since the weights have to be learned as time goes by, we have that \mathbf{W} depends on time and we write $f = f(\mathbf{u}(t, \mathbf{x}), \mathbf{W}(t))$. In the classical approach the penalty term impose a

coherence w.r.t. the Training Set, whereas the regularization term impose the norm of the parameters to be small, so as to f be a smooth function. If we want the process of learning itself to be smooth, we could requires some kind of regularization in the changing of \mathbf{W} during time. We shall see in next sections how this idea could be studied in a *Physics-like* approach.

3 Construction of Cost Functional

In a classical learning problem we have to minimize a cost functional \mathcal{L} w.r.t. $\mathbf{W}(t)$. The functional is composed by a penalty term and a regularization one. The penalty term is calculated on every supervised example, over a training set $\mathcal{P} = \{(u_k, \bar{f}_k)\}_{k=1}^l$, by a loss function $V = f(\mathbf{u}(t, \mathbf{x}), \mathbf{W}(t))$ which is the summation over \mathcal{P} :

$$V(\mathbf{u}(t, \mathbf{x}), \mathbf{W}(t)) = \sum_{k=1}^l \bar{V}(f(\mathbf{u}(t, \mathbf{x}), \mathbf{W}(t)), \bar{f}_k)$$

where \bar{V} could be for example the quadratic function $\bar{V}(f, \bar{f}_k) = \frac{1}{2}(f - \bar{f}_k)^2$ (where f means $f(\mathbf{u}(t, \mathbf{x}), \mathbf{W}(t))$ from now on). Since the examples are presented in time, if t_k is the instant of time in which the couple (u_k, \bar{f}_k) is provide, we can write V as

$$V(\mathbf{u}(t, \mathbf{x}), \mathbf{W}(t)) = \sum_{k=1}^l \bar{V}(f, \bar{f}_k) \cdot H(t - t_k).$$

In a physics-like approach we can look at the term V as the *potential energy*, so as the total energy of the system L is composed by $K + V$ where K is the *kinetic energy* . Then in our formulation we write

$$L = K + \gamma V \tag{1}$$

where γ represent a *regularization parameter* (which includes the classical case when $\gamma = 1$). We can write K as:

$$K = \sum_{i=1}^m \mu_i \dot{\omega}_i^2$$

where m is the total number of weights in \mathbf{W} , and μ_i represent the *mass* of each weight, (i.e. an additional parameters which can be use to choose the

inertia of each weight). Since $\dot{\omega}_i = \frac{d}{dt}\omega_i$ we can replace $D = \frac{d}{dt}$ with a general differential operator T . Now we can finally write our cost functional as

$$\mathcal{S}_\gamma = \int_0^{t_e} \psi(t) \cdot L \, dt. \quad (2)$$

where ψ is a suitable dissipation function, that we take as $\psi(t) = e^{\theta t}$.

4 First Application

4.1 First Order Operator

In our first application of this theoretical framework we analyze the simple case in which f is a linear function of one single real variable $f: \mathbb{R} \rightarrow \mathbb{R}$ and $f = yu + b$ where $u = u(t, x(t)) = x(t)$. In this framework we want to learn the two weights y, b . The problem can be formulated as

$$y^* = \arg \min_{y \in \mathbb{R}} \int_0^{t_e} \psi(t) \cdot L \, dt$$

and analogous formula hold for b .

We start with the case $T = \alpha_0 + \alpha_1 D$. By the application of the *Eulero-Lagrange* equation we have to solve the second order linear differential equation:

$$\ddot{y} + \theta \dot{y} + \beta y - \frac{\gamma}{\mu \alpha_1^2} \sum_{k=1}^l (u_k y_k + b_k - \bar{f}_k) u_k \cdot \delta(t - t_k) = 0 \quad (3)$$

where $\beta = \frac{\alpha_0 \alpha_1 \theta - \alpha_0^2}{\alpha_1^2}$. The solution is then composed by a term given from the homogeneous solution $y^o(t)$ plus a term given from the impulsive response $y^F(t)$. Then we have¹:

$$y(t) = y^o(t) + y^F(t) = y^o(t) + \frac{\gamma}{\mu \alpha_1^2} \sum_{k=1}^l \zeta_k \cdot g(t - t_k) \quad (4)$$

where we posed $\zeta_k = (u_k y_k + b_k - \bar{f}_k) u_k$. For the bias b we have an analogous solution except for the element ζ_k , which represent $\frac{\partial V}{\partial y}$ and in the correspondent formula for b is $\zeta_k = u_k y_k + b_k - \bar{f}_k$.

For the stability of the system during time, we have to impose the *Routh-Hurwitz conditions*, which lead to the relation $\theta > \alpha_0/\alpha_1$.

¹see section 7 for details about practical calculation

4.1.1 Experimental results

A first implementation (in MatLab) of our theoretical results is in the simple case in which we want to approximate a linear function in an interval $[a, b] = [-1, 1]$ of the real axis. We assume that the examples on the training set are equally spaced in time by a factor τ with $t_0 = 0$. This allow us to use a discretization of (4) for the computing of the evolution of the system, as you can see in Section(7.3). We also consider an equally-spaced subdivision of our interval that we cover forward and backward, i.e. we move from a to b and viceversa, so as to guarantee time correlation among the examples. We assign to every point u_k a target $\bar{f}_k = 2 \cdot u_k - 1$, so we desired $y(t) \rightarrow 2$ and $b(t) \rightarrow -1$ after some epochs. For start, we feed the system with a total supervised training set. We studied some results on this first implementation w.r.t. the parameters $\theta, \alpha_0, \alpha_1, \gamma, \mu, \tau$.

If we start our study for a simple case we have a behavior as shown in Fig.1 . We set $\gamma = -1$, $\mu = 1$, $\theta = 5$, $\alpha_0 = 1$, $\alpha_1 = 1$, $y(0) = y'(0) = b(0) = b'(0) = 0$. We repeat the training set for a total of 40 iterations. From the top to the bottom of the figure we can find :

- the plot of the impulse response g (red line)
- the plot of y (blue line) and b (green line)
- the plot of the last 20% of update of the weights
- the plot of the last 10% of update of the weights

Each plot have the same scaling referred to the time t , so as in the graphs is reported the evolution of the system w.r.t. real time (we can think in seconds). The width of each plot depends on both the number of iterations and the parameter τ , but each plot has the same scale in *seconds*. As we can see in Fig.1 the weights have an initial oscillation and then assume a cyclic behavior with a smaller amplitude oscillation when the system is a steady state, near the desired value 2, -1 respectively.

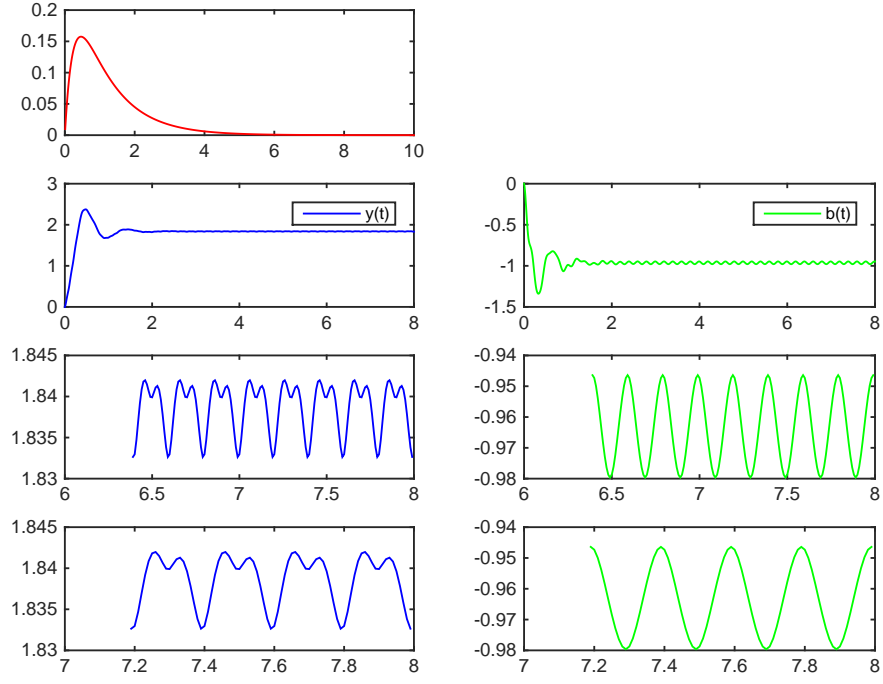


Figure 1: $\gamma = -1$, $\mu = 1$, $\theta = 5$, $\alpha_0 = 1$, $\alpha_1 = 1$, $y(0) = y'(0) = b(0) = b'(0) = 0$, $\tau = 0.01$, iterations = 40.

In the remainder of this section we studied some variations produced by each parameters on the behavior of the weights.

Parameter γ

In our formulation of the theory this parameters only determine the sign between the two terms K and V , so we only explore the set $\{-1, 1\}$ for this one. Because of some correlation with the *gradient descent*, we expect our weights to have a divergent trend when $\gamma = 1$. Our experimental results confirm this, as we can see in Fig.2, where we can notice the trend of the weights after just 5 iterations.

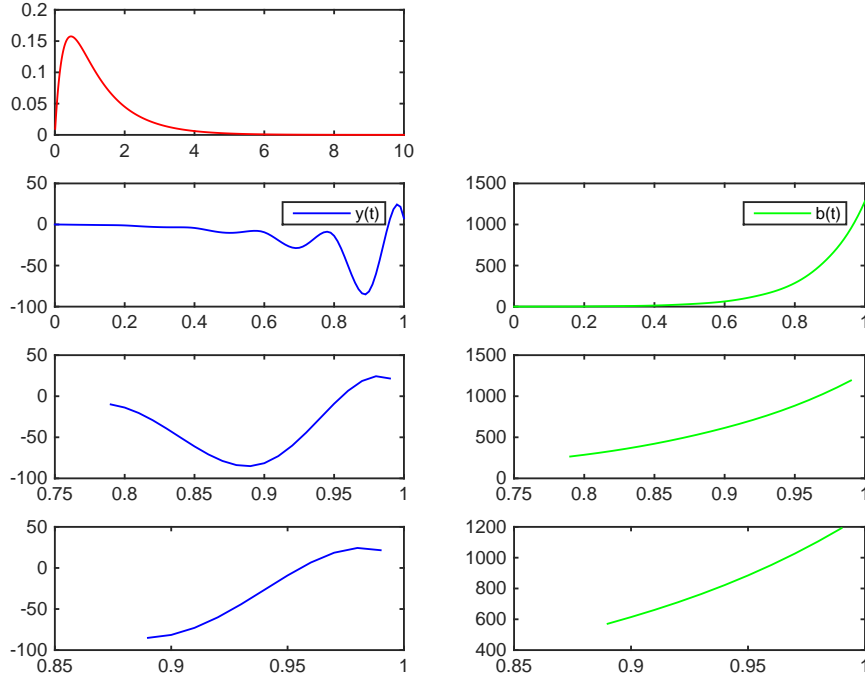


Figure 2: $\gamma = 1$, $\mu = 1$, $\theta = 5$, $\alpha_0 = 1$, $\alpha_1 = 1$, $y(0) = y'(0) = b(0) = b'(0) = 0$, $\tau = 0.01$, iterations = 5.

We tried to vary the other parameters in order to make the weights converge to the desired value also in the case $\gamma = 1$. We obtained converge by imposing a strong regularization by the differential operator

($\alpha_0, \alpha_1 > 10$), but is difficult to find a correlation between the final values and the desired ones(Fig.3). The same result ca be achieved with $\mu > 30$ or $\theta > 120$ and also by increasing the parameter τ . All these adjustments on the other parameters represent the imposition of a strong regularization on the system, which drive all the weights to zero.

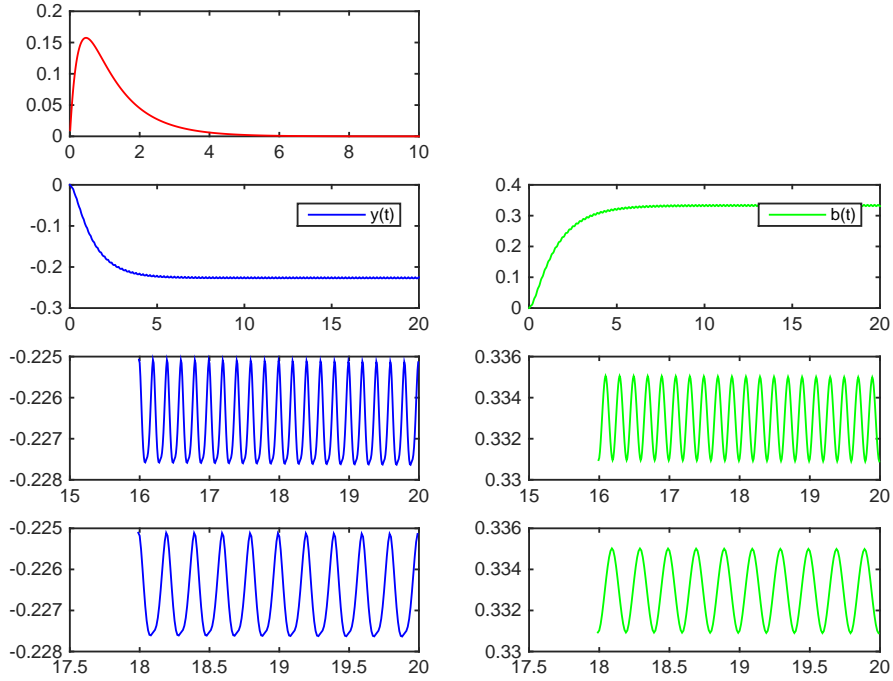


Figure 3: $\gamma=1, \mu=1, \theta=5, \alpha_0=10, \alpha_1=10, y(0)=y'(0)=b(0)=b'(0)=0, \tau=0.01$, iterations =100.

Because of this preliminary results, we drop the parameter γ from now on as we fix $\gamma=-1$.

Parameter τ

This parameters represent the time-sampling step of the system. Under our practical assumptions also the time-spacing of the examples. It represent a crucial parameters of the system and we will study it

after the second order operator. For now we fix $\tau=0.01$

Initial Conditions

Because of the asymptotic behavior of the solution of (3), it is reasonable to assume that different Initial Conditions do not produce relevant changes in the weights at the end of optimization. Indeed in Fig.4 and Fig.5 we can see that the final values of the weights are in the same range of Fig.1, but the initial oscillation is different, due to the different starting points and derivatives. We can also see that for very different initial conditions, the time that the algorithm spent on reach the steady state is almost the same, maybe because the terms ζ_h , which reflect the gradient, balance these differences. As in the previous case, from now on we drop the specification of initial conditions assuming them null.

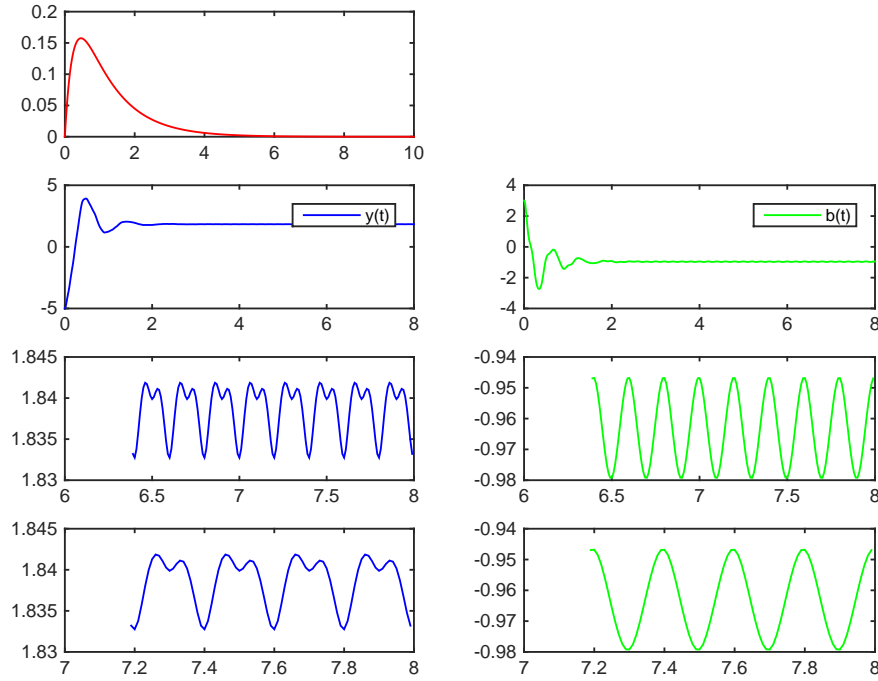


Figure 4: $\mu = 1$, $\theta = 5$, $\alpha_0 = 1$, $\alpha_1 = 1$, $y(0) = -5$, $y'(0) = -2$, $b(0) = 3$, $b'(0) = 4$, iterations = 40.

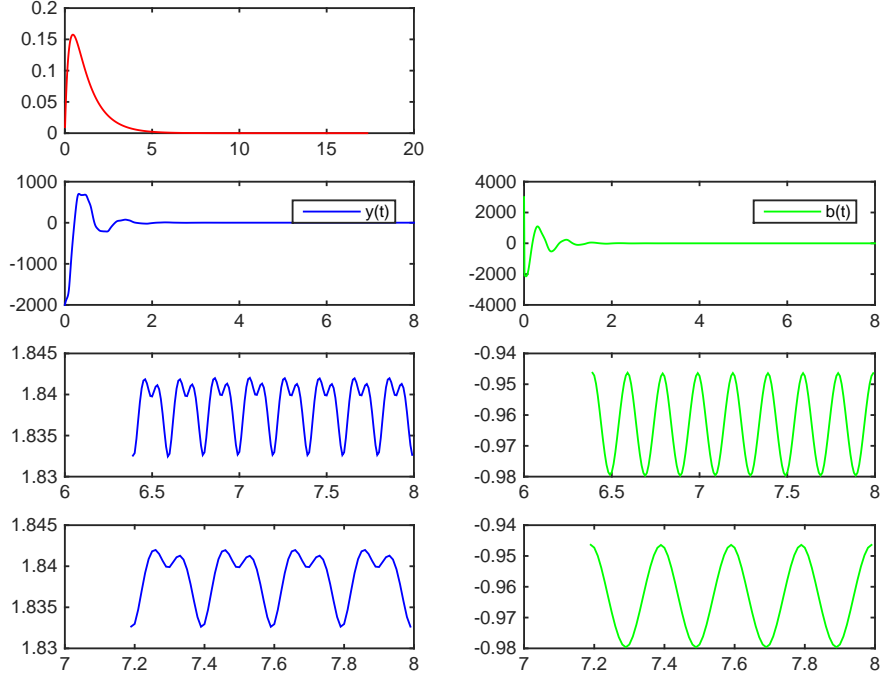


Figure 5: $\mu=1$, $\theta=5$, $\alpha_0=1$, $\alpha_1=1$, $y(0)=-2000$, $y'(0)=-1000$, $b(0)=3000$, $b'(0)=500$, iterations = 40.

Parameter θ

This parameter comes from the exponent in the dissipation term $\psi(t) = e^{\theta t}$ and influences the solution directly in the structure of the functions $g(t)$ and $y^o(t)$ (see section 7). If we decrease $\theta=2$, we have that $g(t) \rightarrow 0$ slowly (Fig. 6), the initial transient phase is longer and oscillations have a greater amplitude. On the opposite, if we set $\theta=10$ we have the reverse effect. In the steady state, we can observe that as θ increases, the average value of the weights decreases ($y(t) \sim 1.956$ for $\theta=2$, $y(t) \sim 1.835$ for $\theta=5$, $y(t) \sim 1.665$ for $\theta=10$), so we can think of a regularization effect.

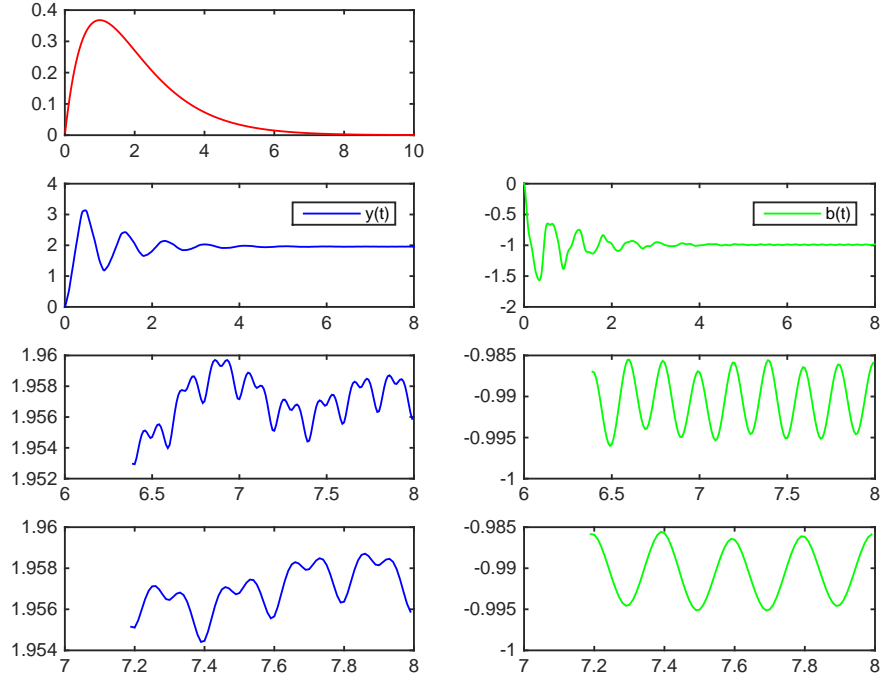


Figure 6: $\mu=1$, $\theta=2$, $\alpha_0=1$, $\alpha_1=1$, iterations = 40.

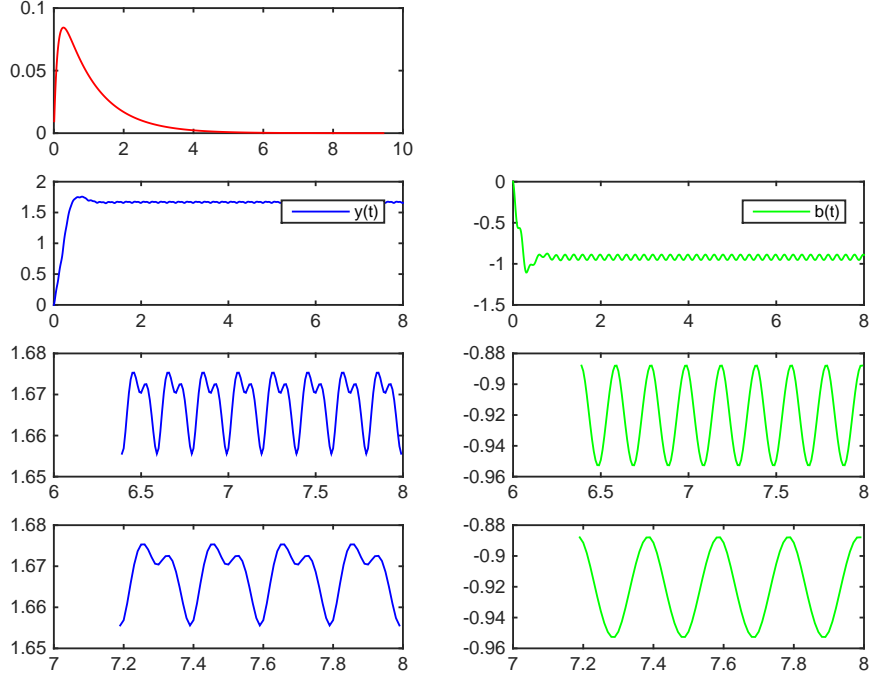


Figure 7: $\mu=1$, $\theta=10$, $\alpha_0=1$, $\alpha_1=1$, iterations = 40.

Parameter α_0, α_1

Also these parameters affect the solution of (3). They are expected to act as a regularization parameters, since they appear directly in the construction of term K . A larger value of α_0 should lead to a weights with a smaller magnitude. α_1 should impose a smaller derivatives, i.e. a smooth variations of the weights during time. These hypothesis are confirmed by Fig.8, Fig.9, Fig.10. In Fig.9, we can notice that a larger α_1 gives not only the expected smoothness, but also a stronger regularization effect w.r.t. α_0 in Fig.8.

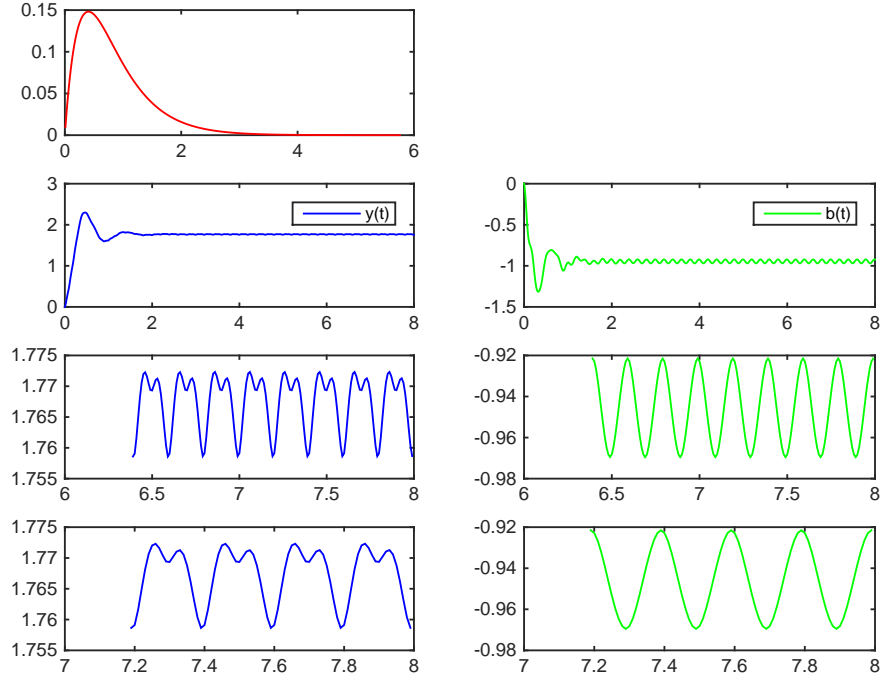


Figure 8: $\mu=1$, $\theta=5$, $\alpha_0=3$, $\alpha_1=1$, iterations = 40.

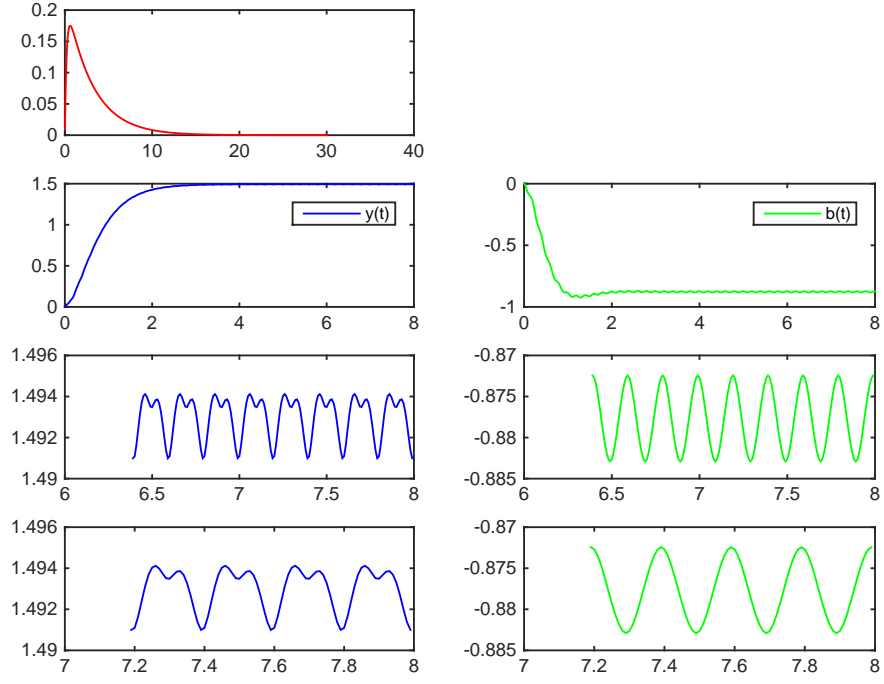


Figure 9: $\mu=1$, $\theta=5$, $\alpha_0=1$, $\alpha_1=3$, iterations = 40.

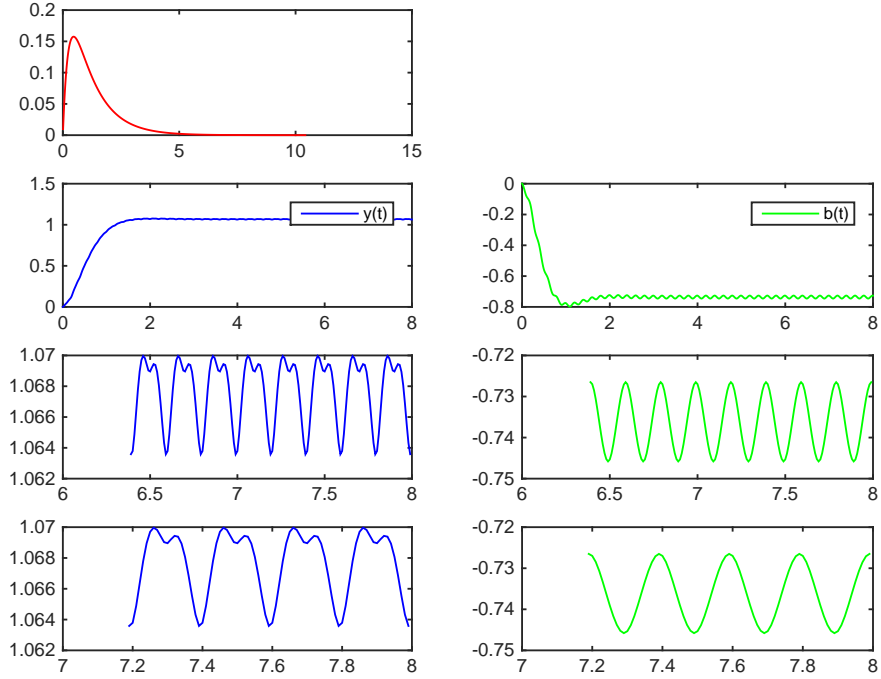


Figure 10: $\mu=1$, $\theta=5$, $\alpha_0=3$, $\alpha_1=3$, iterations = 40.

Parameter μ

μ is the correspondent of the *mass* in physics, so it should represent the inertia of the weights, i.e. how we want to allow to the system to change them at each step (w.r.t. the penalty term V). Furthermore, we can see in (4) that it can be use to represent a sort of learning rate (to follow a parallel with gradient descent again). This is confirmed in Fig.11 and Fig.12 where a smaller value of μ gives more importance to the optimization than to the regularization.

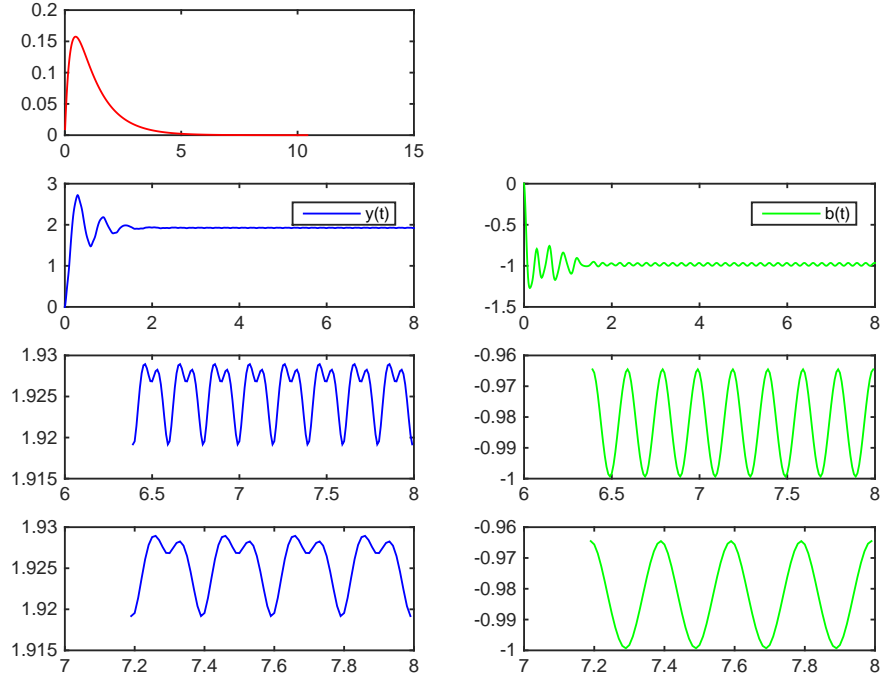


Figure 11: $\mu=0.5$, $\theta=5$, $\alpha_0=1$, $\alpha_1=1$, iterations = 40.

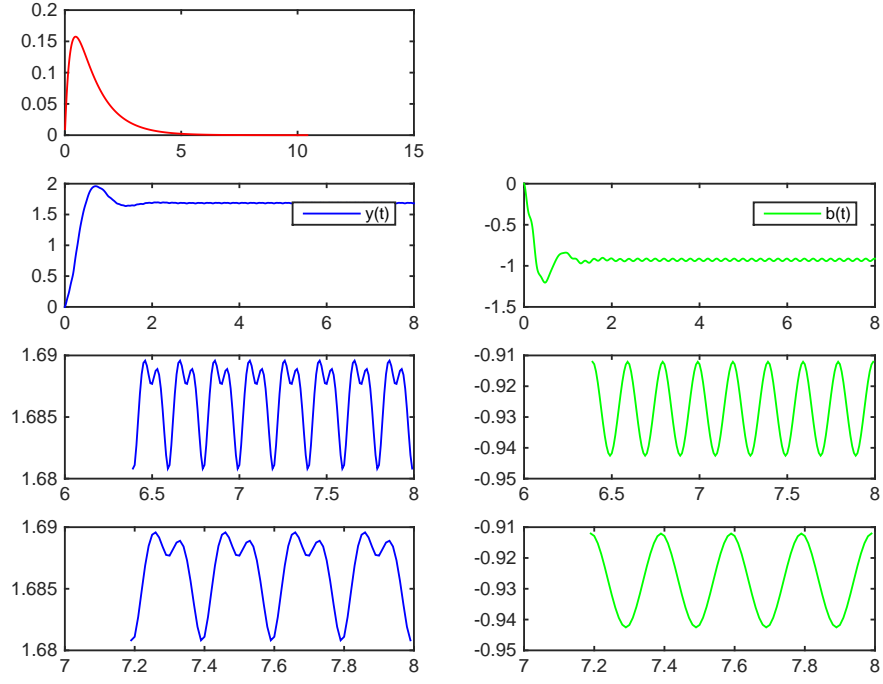


Figure 12: $\mu=2$, $\theta=5$, $\alpha_0=1$, $\alpha_1=1$, iterations = 40.

If we try to grow up with μ , we find that this increment maintain this behavior (unbalancing towards regularization). On the other hand, if we choose a $\mu \leq 0.4$, the system diverges(Fig.13).

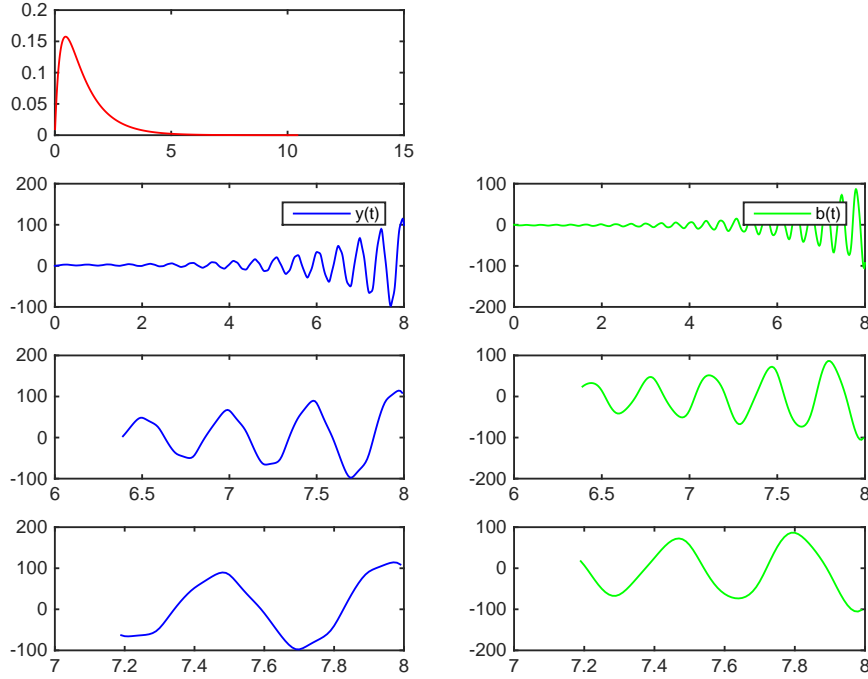


Figure 13: $\mu=0.4$, $\theta=5$, $\alpha_0=1$, $\alpha_1=1$, iterations = 40.

4.1.2 Choice of solutions

All this parameter contribute to model our system, but we have to choose how. Since we would like that the system promptly reacts to the stimuli, we want an Impulsive Response that reach its maximum as fast as possible. Moreover, its useful that the system is width enough to see all the examples more times before to forget them. Once we have chosen a suitable θ (w.r.t. the Training Set, see Section 4.3), we can model the parameters α_j so as to satisfy these request. Often is more convenient to choose the directly the solutions to build our system (see Section 7.2). In Fig.14 we can see that if we chose a solution close to 0, and then the other close to θ (since $\theta = -(\lambda_1 + \lambda_2)$) we have a longer g (green plot), whereas we go in the other direction if the solutions are similar (blue plot).

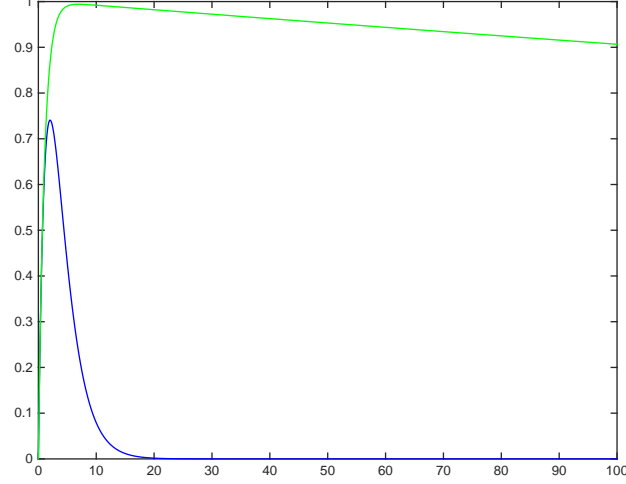


Figure 14: Plot of two different Impulsive Response, the green plot represent g when $\lambda_1 = -0.999$, $\lambda_2 = -0.001$ and the blue one $\lambda_1 = -0.6$, $\lambda_2 = -0.4$

4.2 Second Order Operator

Under the practical assumptions of the previous section, we study the implementation of the case in which the term K is composed by the second order linear differential operator $T = \alpha_0 + \alpha_1 D + \alpha_2 D^2$. The *Eulero-Lagrange* equation lead this time to a fourth order linear differential equation:

$$D^4 y + \beta_3 D^3 y + \beta_2 D^2 y + \beta_1 D y + \beta_0 y + \frac{\gamma}{\mu \alpha_2^2} \sum_{k=1}^l (u_k y_k + b_k - \bar{f}_k) u_k \cdot \delta(t - t_k) = 0 \quad (5)$$

where:

$$\begin{aligned} \beta_0 &= \frac{\alpha_0 \alpha_2 \theta^2 - \alpha_0 \alpha_1 \theta + \alpha_0^2}{\alpha_2^2} \\ \beta_1 &= \frac{\alpha_1 \alpha_2 \theta^2 + (2\alpha_0 \alpha_2 - \alpha_1^2) \theta}{\alpha_2^2} \\ \beta_2 &= \frac{\alpha_2^2 \theta^2 + \alpha_1 \alpha_2 \theta + 2\alpha_0 \alpha_2 - \alpha_1^2}{\alpha_2^2} \\ \beta_3 &= 2\theta \end{aligned} \quad (6)$$

This time the *Routh-Hurwitz conditions* requires:

$$\begin{aligned}
\beta_i &> 0, \quad i = 0, \dots, 3 \\
\beta_3\beta_2 &> \beta_1 \\
\beta_3\beta_2\beta_1 &> \beta_1^2 + \beta_3^2\beta_0
\end{aligned} \tag{7}$$

The updating formula is then²:

$$y(t) = y^o(t) - \frac{\gamma}{\mu \alpha_2^2} \cdot \sum_k^l \zeta_h \cdot g(t - t_k) \tag{8}$$

where we can notice a sign flip before the second term w.r.t. (4) due to the even order of T . This make us expect (because of the parallel with gradient descent again) that this time our system is stable for an opposite sign of γ w.r.t. the first order operator studied in section 4.1.1. In practice, we can observe a more complicated behavior due to the kind of the solutions of (5).

4.2.1 Experimental results

This time we start by observing that there are different possibilities in the kind of the solutions of (5):

(1) Four distinct real solution

We start with the case $\theta = 4$, $\alpha_0 = 0.8$, $\alpha_1 = 1.6$, $\alpha_2 = 0.8$. As we can see in Fig.15 the system is divergent for $\gamma = -1$. Also in the case $\gamma = 1$ in Fig.16 we have divergence, but we can notice a different oscillation of the weights. We can see in (8) that the second term is multiplied by the factor $\gamma/(\mu \alpha_2^2)$. Since $\alpha_2 < 1$ we can apply some of the considerations done in Section 4.1.1 about the parameter μ . This divergence is maybe due to a too higher balancing on the gradients term, indeed if we set $\mu = 4$ we have convergence to the desired values Fig. 17. We can obtain convergence by increasing μ also when $\gamma = -1$ Fig. 18 but with futile values of the weights.

²again see section 7 for details about practical calculation

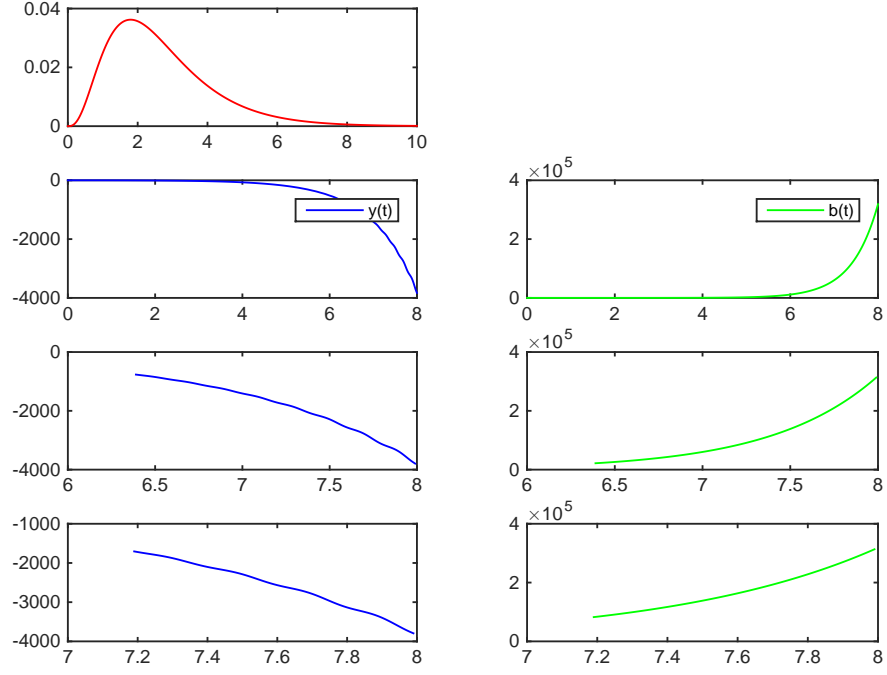


Figure 15: $\gamma = -1$, $\mu = 1$, $\theta = 4$, $\alpha_0 = 0.8$, $\alpha_1 = 1.6$, $\alpha_2 = 0.8$, $\tau = 0.01$, null Initial Conditions , iterations = 40.

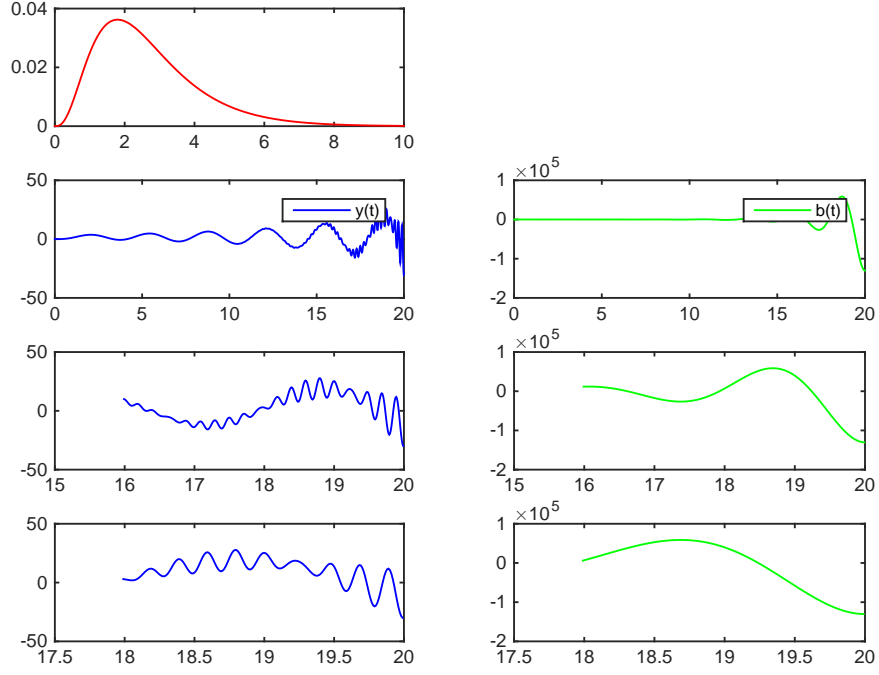


Figure 16: $\gamma=1$, $\mu=1$, $\theta=4$, $\alpha_0=0.8$, $\alpha_1=1.6$, $\alpha_2=0.8$, $\tau=0.01$, null Initial Conditions , iterations=100.

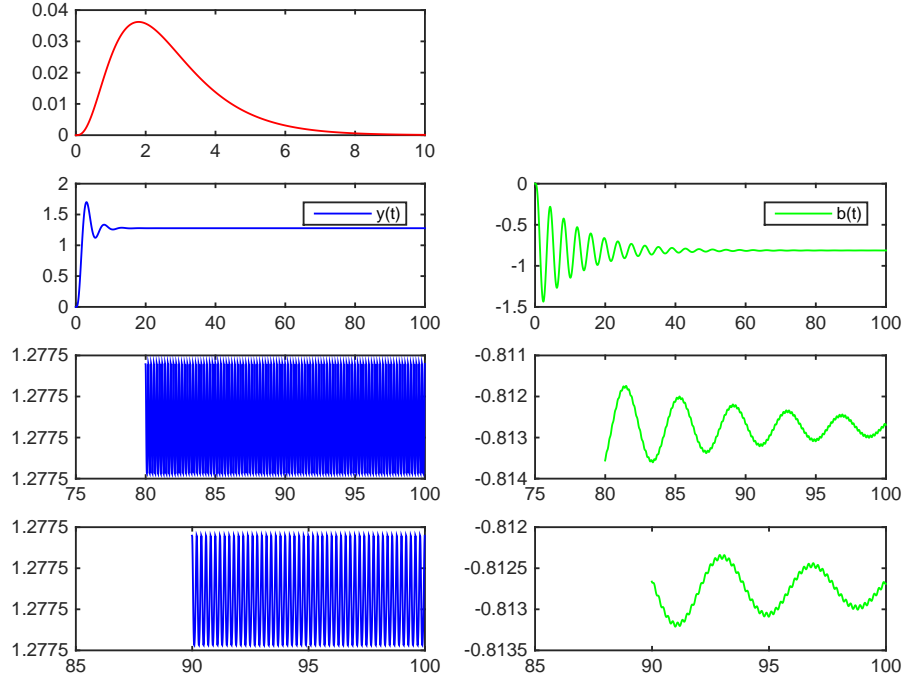


Figure 17: $\gamma=1, \mu=4, \theta=4, \alpha_0=0.8, \alpha_1=1.6, \alpha_2=0.8, \tau=0.01$, null Initial Conditions , iterations=500.

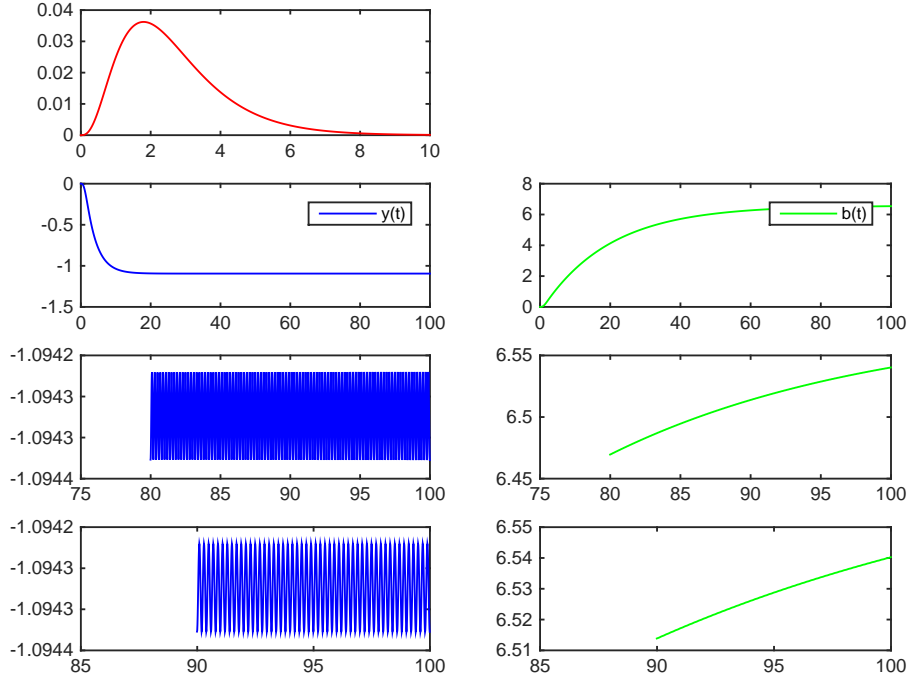


Figure 18: $\gamma = -1$, $\mu = 20$, $\theta = 4$, $\alpha_0 = 0.8$, $\alpha_1 = 1.6$, $\alpha_2 = 0.8$, $\tau = 0.01$, null Initial Conditions, iterations = 500.

We have this kind of solution also for $2.2 \leq \theta \leq 4$ and for $6.6 \leq \theta \leq 7.2$, with the same value of α_j . When $\theta = 2.2$ we need a bigger value of μ to achieve convergence Fig. 19. This is because the impulse response has a bigger maximum than before, and then also the coefficients which multiply the gradients are bigger, so that we need a bigger balancing μ .

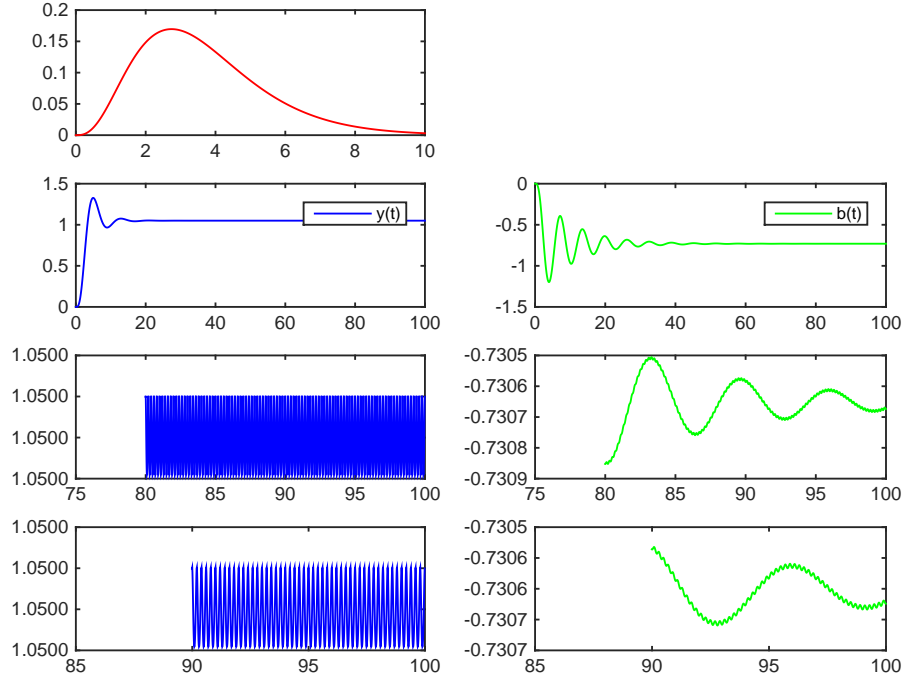


Figure 19: $\gamma = 1$, $\mu = 40$, $\theta = 2.2$, $\alpha_0 = 0.8$, $\alpha_1 = 1.6$, $\alpha_2 = 0.8$, $\tau = 0.01$, null Initial Conditions, iterations = 500.

When $\theta = 6.8$ these first conclusion are confirmed in Fig.20, where $\mu = 1$ is enough for convergence. This is because the solutions to (5) are different, but we can comparing again the impulse response that assume a smaller value than before.

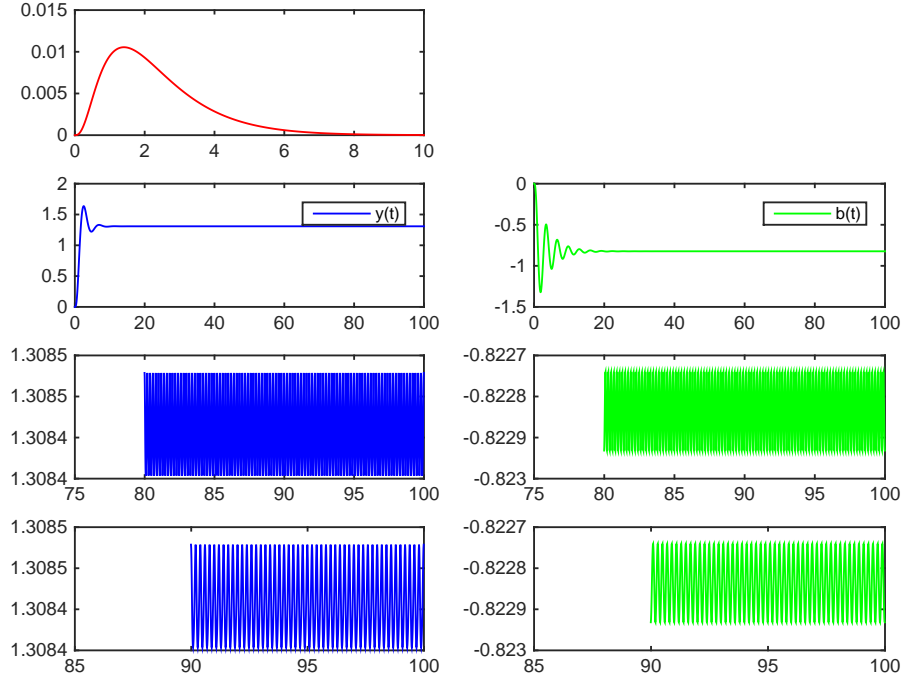


Figure 20: $\gamma=1$, $\mu=1$, $\theta=6.8$, $\alpha_0=0.8$, $\alpha_1=1.6$, $\alpha_2=0.8$, $\tau=0.01$, null Initial Conditions, iterations=500.

(2) Four real solution, 1 with multiplicity 2

Also in this case the system diverge when $\gamma = -1$. When $\gamma = 1$ we report the case in Fig.21 where we can see a behavior similar to case (1).

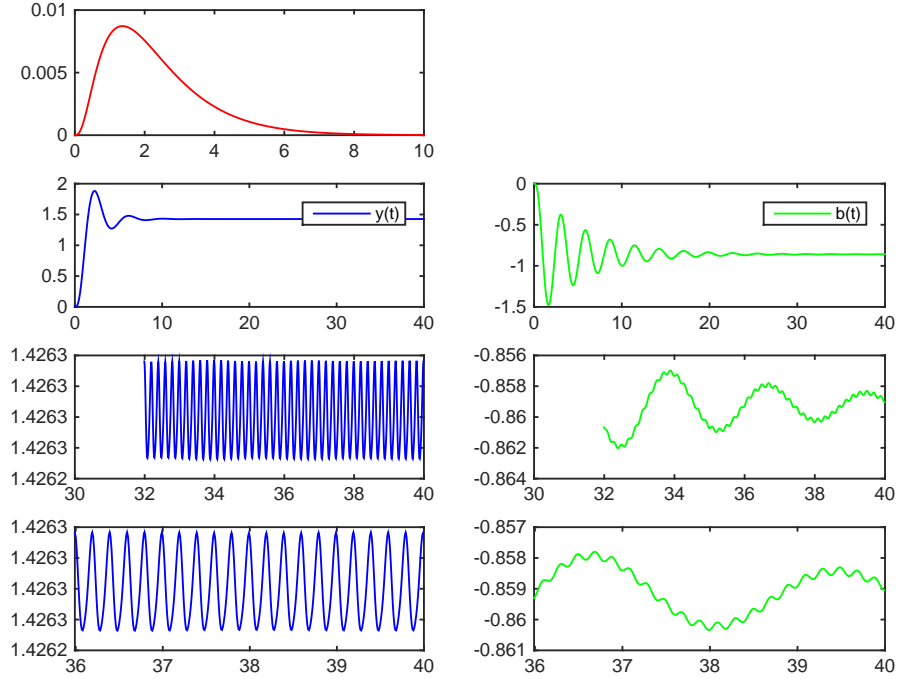


Figure 21: $\gamma = 1$, $\mu = 0.1$, $\theta = 7.4$, $\alpha_0 = 0.2$, $\alpha_1 = 0.4$, $\alpha_2 = 0.2$, $\tau = 0.01$, null Initial Conditions, iterations = 200.

(3) Two distinct real solutions, two conjugate complex solution

We can obtain this solutions for example for these settings of the parameters:

θ	α_0	α_1	α_2
5.75	1	2	1
7	1	2	1
1	1	3	2.25
2.25	1	3	2.25

When we have the complex solution with real part smaller than the real solutions, part of the Impulse Response without oscillation (the one coming from the real solutions) disappears before the other one, and if the imaginary part is big enough we have an oscillation that lead to a more complex behavior and then to instability. We can

choose (see Section 7.2) $\lambda_{1,2} = -0.1 \pm i$, $\lambda_3 = -1.2$, $\lambda_4 = -1$ and we have the situation in Fig.22, where a big regularization is required to convergence.

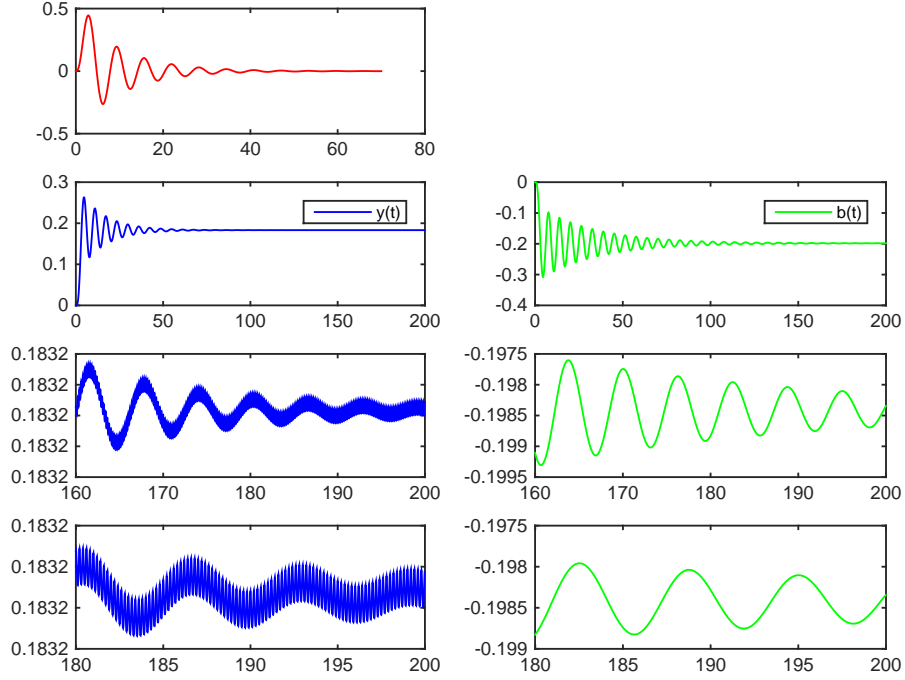


Figure 22: $\gamma = 1$, $\mu = 3000$, Solutions: $\lambda_{1,2} = -0.1 \pm i$, $\lambda_3 = -1.2$, $\lambda_4 = -1$, $\tau = 0.01$, null Initial Conditions, iterations = 1000.

(4) One real solution with multiplicity 2, two conjugate complex solution

We can obtain this solutions for example when:

θ	α_0	α_1	α_2
8.2	0.2	1.2	1.8
18.4	0.2	1.2	1.8

and we have an analogous situation of **(3)**.

(5) Four complex solution, two conjugate pairs

We have both the case in which we have two complex conjugate pairs and the case with a complex conjugate pair with multiplicity two. They have a similar behavior depending on the magnitude of the real and imaginary parts. The real parts influence the memory of the system, whereas the imaginary parts the frequency of the sinusoidal oscillation. In each case is possible to find a value of μ which allow convergence, but often to values near to 0 with an high-oscillatory trend similar to the one reported in Fig.22, since it is due to the sinusoidal nature of the solution.

From this study on the solutions is clear that the parameter μ decides again the balancing between regularization and fitting, as observed for the first order operator. The behavior w.r.t. Initial Conditions is again the same as we can see in Fig. 24. Since we have a fourth-order differential equation we need to know the values of the first $n - 1 = 3$ derivatives of the solution $y^o(t)$ in $t=0$. We indicate I.C. with the vectors $y_0 = [y^o(0), y^{o(1)}(0), y^{o(2)}(0), y^{o(3)}(0)]$ and $b_0 = [b^o(0), b^{o(1)}(0), b^{o(2)}(0), b^{o(3)}(0)]$.

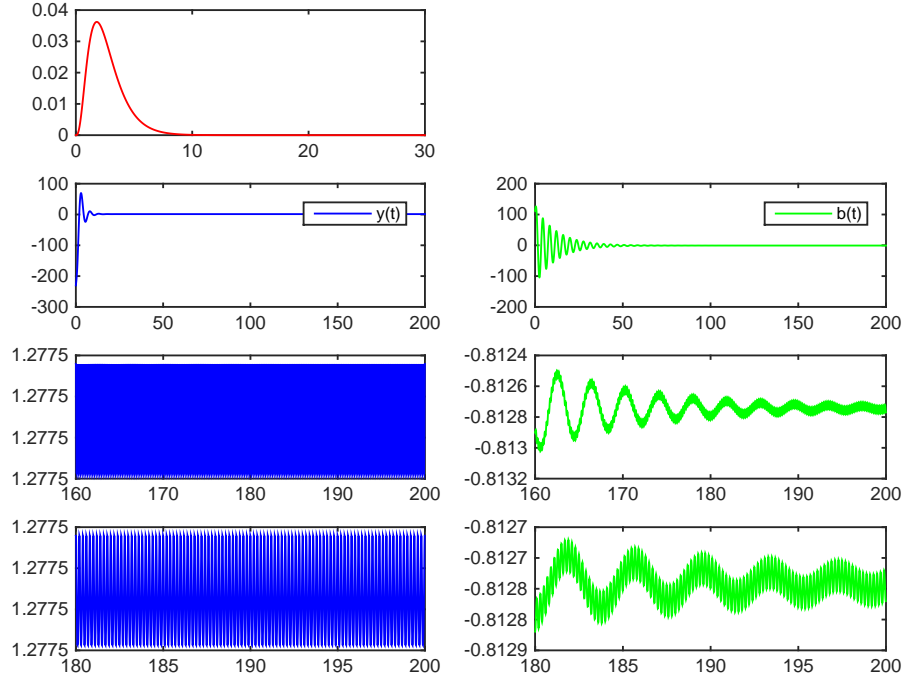


Figure 23: $\gamma=1$, $\mu=4$, $\theta=4$, $\alpha_0=0.8$, $\alpha_1=1.6$, $\alpha_2=0.8$, $\tau=0.01$, $y_0 = [-230, 54, 0, -342]$, $b_0=[112, 41, 19, 0]$, iterations = 1000.

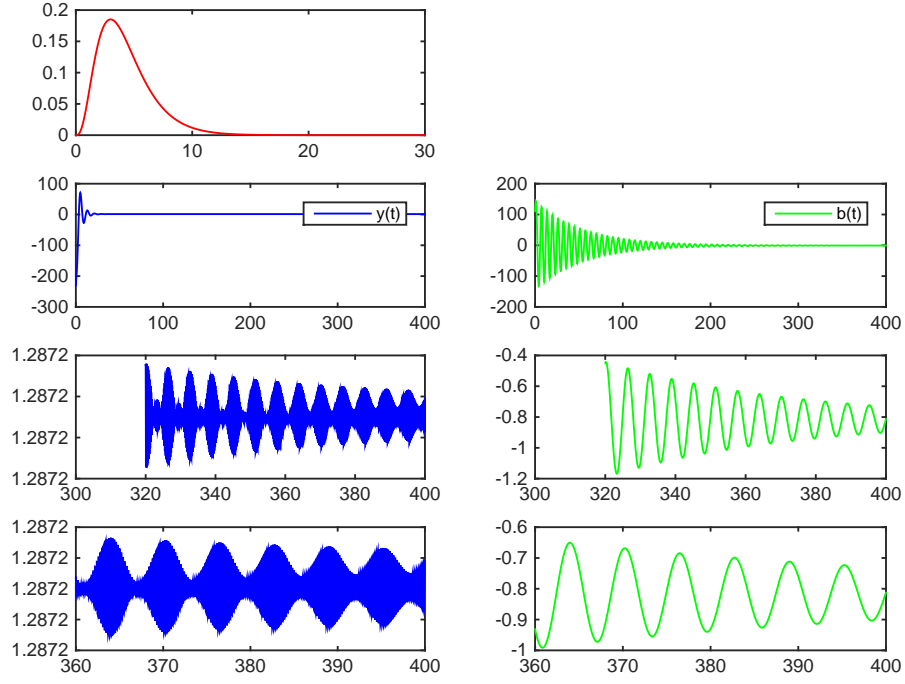


Figure 24: $\gamma=1$, $\mu=4$, $\theta=2.25$, $\alpha_0=1$, $\alpha_1=3$, $\alpha_2=2.25$, $\tau=0.01$, $y_0 = [-230, 54, 0, -342]$, $b_0=[112, 41, 19, 0]$, iterations=2000.

4.2.2 Choice of solutions

Like in the First order case, we are allowed to choose a suitable set of solutions and then find the parameters for our model (Section 7.2). In this case we have four solutions related to θ by the relation $2\theta = -(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)$. Also in this case we need a solution close to 0 to allow memorization. It is also useful not to choose another one or two little (w.r.t. θ) solutions since this makes the g grow too much (Fig.25).

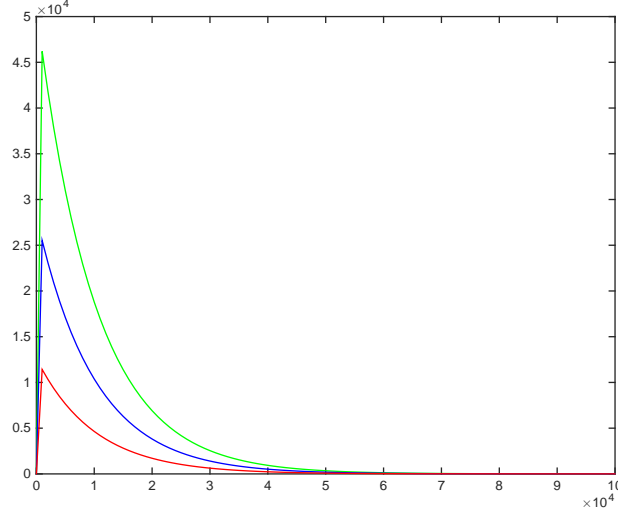


Figure 25: We have in the green plot $l_1 = -0.0599$, $l_2 = -0.06$, $l_3 = -0.01$, $l_4 = -0.0001$, in the blue one $l_1 = -0.0199$, $l_2 = -0.1$, $l_3 = -0.01$, $l_4 = -0.0001$ and in the red one $l_1 = -0.0399$, $l_2 = -0.04$, $l_3 = -0.05$, $l_4 = -0.0001$.

4.3 Sampling-step τ , Parameter θ and number of Impulses

In our first experiments we consider a totally supervised Training set, where the examples are equally spaced both in time and space. The first example comes at $t_1 = \tau$, then the first supervision came at $\frac{3}{2}\tau$ (see Section 7.3) and the system receives an impulse. The next example comes τ seconds after the first and so on. Since the memory is related to the saturation time of impulse response, the learning process of the system embraces all the examples appearing in the interval of time before this saturation. This means that the system has to be build so as the saturation time comes after the whole Training Set has been seen more times. Further more, since the functional in (2) contain the term $e^{\theta t}$, the parameter θ has to be such that $e^{\theta t_0} = 1$ is not too much smaller than $e^{\theta t_l}$ (t_l instant at which the last example u_l comes). Another important characteristic to take in account is the delay the Impulse Response. If the supervisions (and then the impulses themselves) are too frequent, there is an accumulation of this delay that could cause instability. For these reasons it is important to study the behavior of the system w.r.t. the rate between θ and τ , with some adjustment allowed by

the other parameters. In this section its convenient to restrict the analysis to the second order differential operator by managing the solutions of the differential equations (5). The parameter θ is directly related to the roots λ by the relation

$$2\theta = -(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4).$$

This allow us to choose a suitable θ for our model by the solutions, then find the other parameters of the model to optimize the behavior (see Section 7.2). One solution close to 0 gives memory to our system. So we choose λ_1 small enough to give sufficient memory w.r.t. data, from now on we fix $\lambda_1 = -1 \times 10^{-8}$. We split $\theta - \lambda_1$ roughly equally among the others solutions, since in this way we have a quicker response with a smaller maximum (see Section 7.2), we assume they are respectively the 60,65 and 75% of 2θ . Because of this choice on the parameters, in the following figures we plot at the top both the behavior of function g near the origin and its global trend. We also pose $\eta = \gamma/\mu$. At the bottom we plot the last five iteration on the Training Set, to better see how the oscillation at the steady state depends on the data set. In the label we specify SO to indicate we are dealing with the second order operator.

As a first experiment we try to better understand the oscillatory behavior when the system reach the steady state. For start we choose our models. We start with the same Training set with $l=20$ and $\tau=0.01$, $\theta=1$ produce $e^{\theta t_l} = 1.22$, using the parameters reported in Fig.26 and referring to this configuration as the *low* dissipation one.

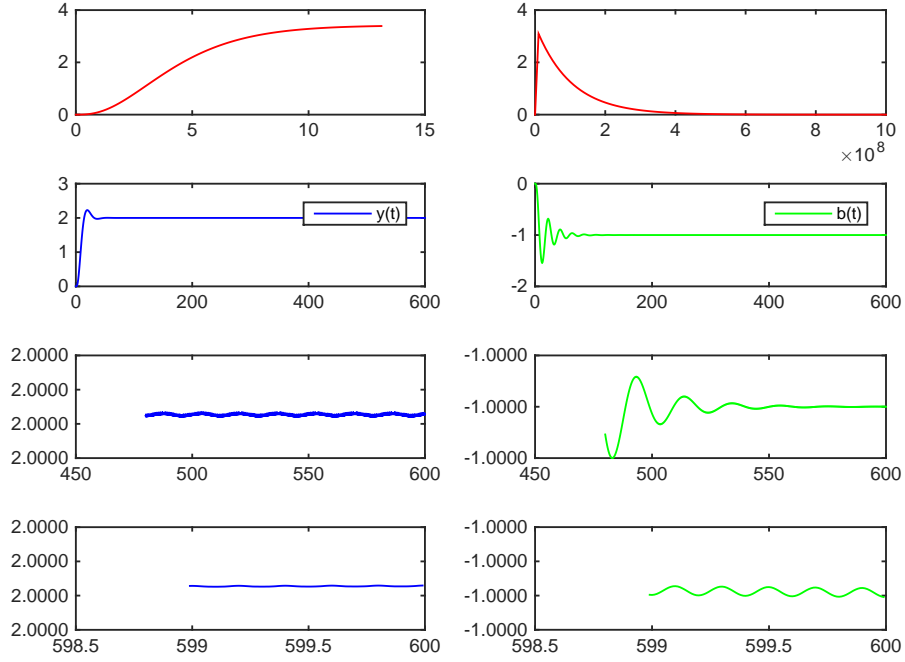


Figure 26: SO: $\eta=0.001$, $\theta=1$, $\tau=0.01$, iterations = 3000.

When we increase the parameter $\tau = 0.1$, also the period T and the oscillation period have the same increment, as shown in Fig.27.

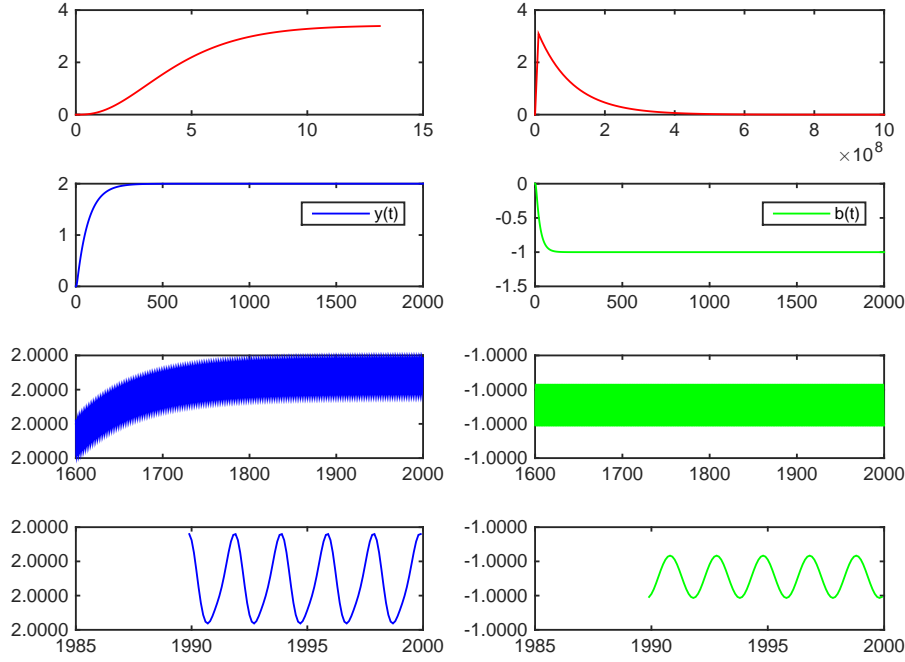


Figure 27: SO: $\eta=0.001$, $\theta=1$, $\tau=0.1$, iterations = 2000.

Now we show the behavior of the system when we reproduce a situation similar to the classic *Stochastic Gradient Descent*. Since the memory of the system is very long w.r.t. data (because of λ_1), the function g is almost constant once it reach its maximum. If the examples are far enough to allow the system to respond between two of them (i.e. g reach its maximum), we reply the gradient descent algorithm. Each examples modify the weights with a term related to the gradient calculated at the previous instant of time. As already said, the period both depends on τ and θ , so that we can arbitrarily fix $\theta = 1$ and enlarge $\tau = 40$ so as to obtain the desired configuration with $e^{\theta t_i} = 10^{173}$. The outcome of this *high* dissipation setting is showed in Fig.28. In the remainder of this Section we can see at the top of the figures g again, then the behavior of the two weights y (blue) and b (green) after the first iteration on the Training Set, the total trend and the last 5 iterations at the bottom.

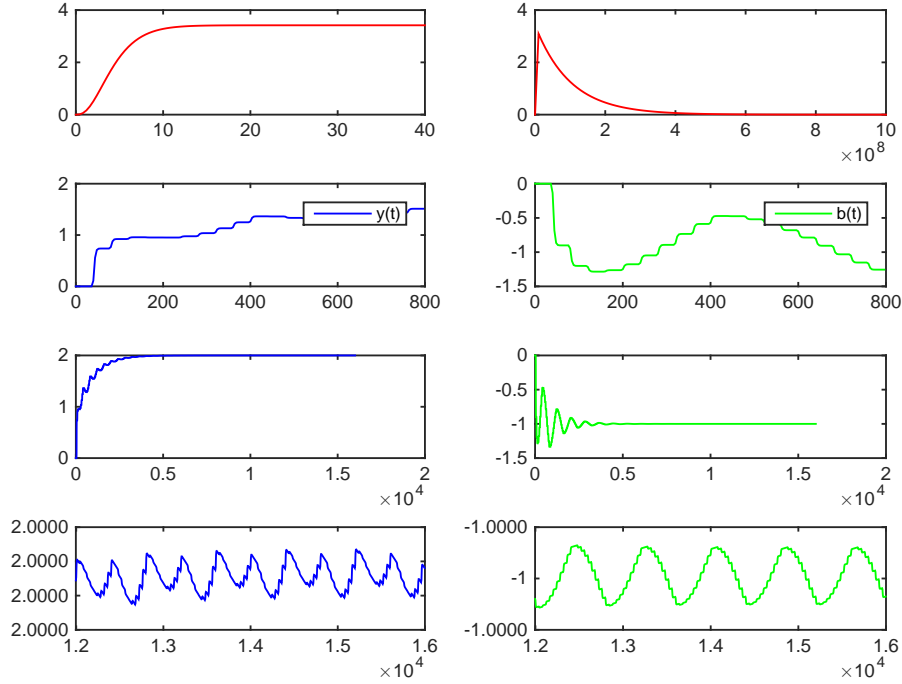


Figure 28: SO: $\eta=0.1$, $\theta=1$, $\tau=40$, iterations = 20.

5 First application on ANNs

As first simple practical application we try some experiment in the optimization of a simply ANN. We use a network with one hidden layer and an output layer, the identity as output function and the *rectifier* function:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

as activation function. In this model we have simply to extend the updating formulas for the weights y , b to the weights of the two layers. In this first application we try different setting of the parameters, with different number of units and for both the first and the second order differential operator. In the next list of experiments, we refer to some results obtained by use $\theta=1$ and 20 of units in the hidden layer, τ varying so as to guarantee a good

value of $e^{\theta\tau l}$, η has been changed to guarantee the best fitting, in the second order differential operator case.

5.1 One dimension functions

We first attack the practical model of the first sections, i.e. a regression task on a Set containing 100 points $u_k \in [-1, 1]$, which are sorted and equally spaced. All the points are labeled with the target $y_k = 2 \cdot u_k - 1$, but only 10 points give supervision. We have the $MSE = 1.77 \cdot 10^{-3}$ after $2 \cdot 10^4$ iterations. In Fig.29 we can see the trend of MSE in other $2 \cdot 10^5$ epochs if we turn off the supervisions (only labeled points for MSE evaluation).

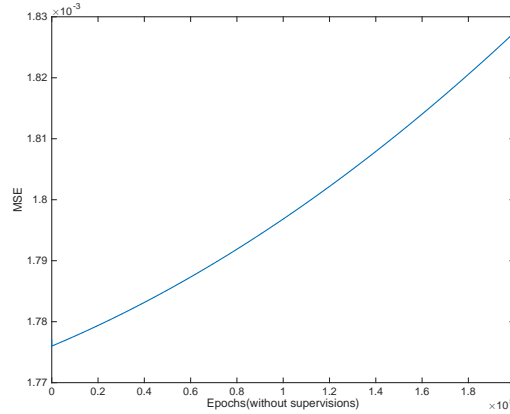


Figure 29: Trend of MSE in $2 \cdot 10^5$ iterations without supervisions in regression task. Parameters of the system: $\eta = 10^{-4}$, $\theta = 1$, $\tau = 0.01$, Units: 20.

We then try the same parameters for a classification task on the same set. We assign the target class *true* ($\tilde{f}_k = [1 \ 0]'$) to the points in $[-0.5, 0.5]$ and the class *false* ($\tilde{f}_k = [0 \ 1]'$) to the others. Again we use only 10 points for supervision. After $5 \cdot 10^4$ iterations we have $MSE = 0.03$ and $Accuracy = 0.97$. Again we try to turn off the supervisions and go on with agent for other $2 \cdot 10^5$ epochs. Both MSE and accuracy remain almost the same.

5.2 Two dimensions functions

We choose our point in $[-1, 1] \times [-1, 1]$. We use two different trajectories to cover the Training set, a spiral and a flower. We obtain the points of the

spiral as:

$$\mathbf{u}(t) = \begin{cases} (t/100) \cos(t) \\ (t/100) \sin(t) \end{cases}$$

whereas the flower trajectory is obtained by

$$\mathbf{u}(t) = \begin{cases} \cos(10t) \cdot \cos(t) \\ \cos(10t) \cdot \sin(t) \end{cases}$$

We take 100 supervised points coming from each trajectories with $t = 1, \dots, 100$ (26 and 40 for the class true, respectively for the flower and spiral trajectory). The points in $\{(x, y) \in \mathbb{R}^2 : |x| + |y| \leq 0.5\}$ represent the class *true*, the others are false. We divide our experiments in two different phases. In the first phase, we train the network with the supervised points for 10^5 iterations on the set obtained with one trajectory. In the second phase (validation) we check the performance of the system after some epochs without supervisions. We evaluate the performance on the two sets coming from different trajectories and on set obtained by an equally spaced grid of $[-0.5, 0.5] \times [-0.5, 0.5]$ containing 100 examples, equally split in true and false label. The results are in Table 1.

Training trajectory	θ	τ	η	$e^{\theta\tau l}$	Units	Accuracy			
						Training	Validation (no supervisions)		Set
						10^5 epochs 10^4 sec.	10^3 epochs 10^5 sec. ($\tau=1$)	$2 \cdot 10^3$ epochs 10^7 sec. ($\tau=100$)	
Spiral	1	0.001	10^{-4}	1.10	20	0.96	0.71	0.58	Spiral
						0.95	0.63	0.53	Flower
						0.81	0.42	0.40	Grid
						0.98	0.40	0.40	Spiral
Flower						0.99	0.26	0.26	Flower
						0.85	0.40	0.40	Grid

Table 1: First 2D classification results, Training phase with 100 supervised points covered with different trajectories

5.3 Vowels Classifications

We record few tracks with the sequential pronunciation of the five Italian vowels. We process the files with Matlab and take the auditory spectra coefficients coming from RASTA PLP algorithm. We obtain a sampling of the tracks with points in a 40 dimensions features space. Set 1 is obtained from a 20 seconds track with sequential pronunciation of the vowels(2053 samples, 700 labeled). Set 2 (2144 samples) is obtained from a 20 seconds track with sequential pronunciation of the vowels repeated in time (600 labeled

points). Set 3 is obtained from 5 tracks, each containing the pronunciation of a vowel (14934 labeled points). We carry on the experiments with the same approach of the previous section. In the first phase we train the net with a few supervised samples. Again, in the second phase we turn off the supervisions and study the performance of the system as time goes by. After some epochs, we evaluate the agent on each set. The results are in Table 2.

Supervisions	θ	τ	η	$e^{\theta\tau l}$	Units	Accuracy			
						Training 10 ⁵ epochs 10 ⁵ epochs	Validation (no supervisions)		Set
							10 ³ epochs 2.053·10 ⁶ sec. ($\tau=1$)	2·10 ³ epochs 2.053·10 ⁷ sec. ($\tau=10$)	
30 from Set1	1	0.01	10 ⁻⁵	1.34	20	1	1	0.99	Set1
						0.82	0.82	0.72	Set2
						0.84	0.84	0.82	Set3
					100	1	1	0.95	Set1
						0.79	0.78	0.64	Set2
						0.83	0.83	0.77	Set3
30 from Set1 30 from Set2	1	0.005	10 ⁻⁵	1.34	20	1	1	1	Set1
						0.90	0.91	0.93	Set2
						0.87	0.87	0.87	Set3
					100	1	1	1	Set1
						0.94	0.94	0.94	Set2
						0.90	0.90	0.90	Set3
30 from Set1 30 from Set2 30 from Set3	1	0.003	10 ⁻⁵	1.31	20	1	1	1	Set1
						0.95	0.95	0.93	Set2
						0.96	0.96	0.95	Set3
					100	1	1	1	Set1
						0.95	0.95	0.95	Set2
						0.96	0.96	0.96	Set3

Table 2: First Vowels classification results

6 Conclusions

As already said, the studied applications are not the perfect suit of our theory. However, the positive results showed could strengthen our hypothesis and help to better understand the meaning of the different aspects. This made us look for a deeper analysis from many theoretical points of view. We are talking about study others differential operators, cost functionals and forms of the function f . Simultaneously, we would like to investigate the behavior of the current model in applications in which a manifold regularization in time plays an fundamental role, as in Computer Vision problems.

7 Appendix

7.1 General solution and coefficients

In this section we report some practical calculations and assumptions to solve the differential equation of our theoretical framework. The notation is finalized to a practical general implementation.

We can write the general form of characteristic polynomial as:

$$\beta_n \lambda^n + \dots + \beta_1 \lambda + \beta_0 = 0 \quad (9)$$

Then we have a set of J solution λ_j each with they multiplicity r_j . The Laplace Transform lead to:

$$G(s) = \frac{1}{\sum_{q=0}^n \beta_q \lambda^q} = \sum_{j=1}^J \sum_{i=1}^{r_j} \frac{c_{ji}}{(s - \lambda_j)^{r_i}} \quad (10)$$

Where c_{ji} are constants such that:

$$\sum_{j=1}^J \sum_{i=1}^{r_j} c_{ji} (s - \lambda_j)^{r_j-i} \left(\prod_{\substack{k=1 \\ k \neq i}}^J (s - \lambda_k)^{r_k} \right) = 1 \quad (11)$$

For practical issue we pose

$$\begin{aligned} \Lambda^{j,j} &= \begin{bmatrix} \lambda_1 & \dots & \lambda_1 & \dots & \lambda_J & \dots & \lambda_J & \lambda_j & \dots & \lambda_j & \end{bmatrix} \\ R^{j,j} &= \begin{bmatrix} 1 & \dots & r_1 & \dots & 1 & \dots & r_J & 1 & \dots & r_j & \end{bmatrix} \\ \Lambda^{j,q} &= \begin{bmatrix} \lambda_1 & \dots & \lambda_1 & \dots & \lambda_J & \dots & \lambda_J & \lambda_j & \dots & \lambda_q & \end{bmatrix} \\ R^{j,q} &= \begin{bmatrix} 1 & \dots & r_1 & \dots & 1 & \dots & r_J & 1 & \dots & r_q & \end{bmatrix}, \quad q \leq j \\ I_n &= \begin{bmatrix} 1 & \dots & n & \end{bmatrix} \end{aligned} \quad (12)$$

To simplify the notation we pose $R = R^{J,J}$, $\Lambda = \Lambda^{J,J}$. If we carry out the summation in (10) (with the new notation), pose $n_j = n - R_j$ we find that each c_{ji} multiply a factor:

$$c_{ji} \left[\sum_{k=0}^{n_j} s^{n_j-k} \left(\sum_{i \in C_{n_j,k}(I_{n_j})} \left(\prod_{p=i_1}^{i_{n_j}} -\Lambda_p^{j,i} \right) \right) \right] = c_{ji} \sum_{k=0}^{n_j} s^{n_j-k} A_{l,j+i}, \quad l = n_j - k + 1 \quad (13)$$

then we can determine $C = [c_{11} \cdots c_{1r_1} \cdots c_{J1} \cdots c_{Jr_J}]'$ from the system $AC = b$, $b = [10 \cdots 0]'$ and $A \in \mathbb{R}^{n,n}$ with:

$$A_{l,j+i} = \begin{cases} \sum_{k \in C_{n_j,k}(I_{n_j})} (\prod_{p=i_1}^{i_{n_j}} -\Lambda_p^{j,i}) & \text{if } 1 \leq l \leq n_j + 1, k = n_j + 1 - l \\ 0 & n_j + 1 < l \leq n \end{cases} \quad (14)$$

The general solution is of the form:

$$g(t) = \sum_{j=1}^J C_j t^{R_j-1} e^{\Lambda_j t} \quad (15)$$

When we have a complex solution $\lambda_j = \alpha + i\beta$, also its conjugate $\lambda_p = \bar{\lambda}_j = \alpha - i\beta$ is present and also the relatives constants are such that $C_p = \bar{C}_j = \alpha_c - i\beta_c$. We have in the solution :

$$\begin{aligned} & \cdots + C_j \cdot e^{\alpha t} (\cos \beta t + i \sin \beta t) + C_p \cdot e^{\alpha t} (\cos(-\beta t) + i \sin(-\beta t)) + \cdots \\ & \cdots + C_j \cdot e^{\alpha t} (\cos \beta t + i \sin \beta t) + C_p \cdot e^{\alpha t} (\cos(\beta t) - i \sin(\beta t)) + \cdots \\ & \cdots + (C_j + C_p) \cdot e^{\alpha t} (\cos \beta t) + (C_j - C_p) \cdot e^{\alpha t} (i \sin \beta t) + \cdots \end{aligned} \quad (16)$$

and since

$$\begin{aligned} (C_j + C_p) &= \alpha_c + i\beta_c + \alpha_c - i\beta_c \\ (C_j - C_p) &= \alpha_c + i\beta_c - \alpha_c + i\beta_c \end{aligned}$$

the (16) becomes

$$\begin{aligned} & \cdots + 2\alpha_c \cdot e^{\alpha t} (\cos \beta t) + 2i\beta_c \cdot e^{\alpha t} (i \sin \beta t) + \cdots \\ & \cdots + 2\alpha_c \cdot e^{\alpha t} (\cos \beta t) - 2\beta_c \cdot e^{\alpha t} (\sin \beta t) + \cdots \end{aligned} \quad (17)$$

since this is the contribution of two solution with the same *real* and *imaginary* parts, we can write (15) as

$$g(t) = \sum_{j=1}^J t^{R_j-1} e^{\Re(\Lambda_j)t} (\Re(C_j)(\cos(\Im(\Lambda_j)t)) - \Im(C_j)(\sin(\Im(\Lambda_j)t))) \quad (18)$$

also the solution to the homogeneous equation is of the form

$$y^o(t) = \sum_{j=1}^J K_j t^{R_j-1} e^{\Lambda_j t} \quad (19)$$

where K_j are determined by imposing the Initial Conditions given for $y^o(t)$, that is $Y_0 = [y^o(0) \cdots y^{o(n-1)}(0)]'$. Since

$$y^{o(d)}(0) = \sum_{j=1}^J (d+2-R_j)^+ K_j \Lambda_j^{(d+1-R_j)^+} \quad (20)$$

we can find $K = [K_1 \cdots K_n]'$ by solving the system $MK = Y_0$ where:

$$M_{vj} = (v+1-R_j)^+ \Lambda_j^{(v-R_j)^+} \quad (21)$$

and exactly as in the case of $g(t)$ we can write

$$y(t) = \sum_{j=1}^J t^{R_j-1} e^{\Re(\Lambda_j)t} (\Re(K_j)(\cos(\Im(\Lambda_j)t)) - \Im(K_j)(\sin(\Im(\Lambda_j)t))) \quad (22)$$

7.2 From solution to parameters

In section 4.2 we saw that for the second order case convergence depends not only on γ , but also on the kind of the solutions of the characteristic polynomial, i.e. on β_j . Moreover, the most important parameter of our model is θ , which allow to choose the memory width of the system. This memory is related to the function $\psi(t) = e^{\theta t}$, which represent the weight that the model assigns to each samples as the time goes by. The smaller is θ (but always > 0) the bigger is the memory of our model. We are interested in find suitable values of α_j which allow convergence when θ is small. In practice, we can build our model only by the solutions of the characteristic poly, since the parameters influence the updating formulas (4),(8) only by the last α_j , which can be absorbed in the term μ . Then we can choose directly suitable solutions, and verify that they are related to meaningful values of parameters. So, starting from the solutions (chosen in a way to have the desired θ), is easy to find the coefficients of the characteristic poly and then find the rate between the α_j that generate the model.

7.2.1 First Order

If we have the solutions λ_1, λ_2 the characteristic poly of (3) is

$$\lambda^2 + \theta\lambda + \beta = (\lambda - \lambda_1)(\lambda - \lambda_2).$$

So the value of θ is given by $\theta = -(\lambda_2 + \lambda_1)$ and the rate $\frac{\alpha_0}{\alpha_1}$ is computable from β . That is, if we pose $\nu = \frac{\alpha_0}{\alpha_1}$, we can find suitable value of α_j from the solutions of

$$\nu^2 - \theta\nu + \beta = 0. \quad (23)$$

7.2.2 Second Order

In this case the characteristic poly is $\lambda^4 + \beta_3\lambda^3 + \beta_2\lambda^2 + \beta_1\lambda + \beta_0 = 0$. The coefficient β_3 is still given by the opposite of the summation among the solutions, and since $\beta_3 = 2\theta$ it is still easy to set θ . To find the rates among α_j is convenient to work with $\nu_0 = \frac{\alpha_0}{\alpha_2}$ and $\nu_1 = \frac{\alpha_1}{\alpha_2}$ so that:

$$\beta_0 = \frac{\alpha_0\alpha_2\theta^2 - \alpha_0\alpha_1\theta + \alpha_0^2}{\alpha_2^2} = \nu_0\theta^2 - \nu_0\nu_1\theta + \nu_0^2 \quad (24)$$

$$\beta_1 = \frac{\alpha_1\alpha_2\theta^2 + (2\alpha_0\alpha_2 - \alpha_1^2)\theta}{\alpha_2^2} = \nu_1\theta^2 + 2\nu_0\theta - \nu_1^2\theta \quad (25)$$

$$\beta_2 = \frac{\alpha_2^2\theta^2 + \alpha_1\alpha_2\theta + 2\alpha_0\alpha_2 - \alpha_1^2}{\alpha_2^2} = \theta^2 + \nu_1\theta + 2\nu_0 - \nu_1^2 \quad (26)$$

From the equation (26) we can find the relation among the coefficients:

$$\beta_1 = \frac{\beta_3\beta_2}{2} - \frac{\beta_3^3}{8} \quad (27)$$

from equation (25) we find

$$\nu_0 = \frac{1}{2\theta} (\beta_1 + \nu_1^2\theta - \nu_1\theta^2) \quad (28)$$

and from (24) we have

$$\nu_1^4 - 4\theta\nu_1^3 + \nu_1^2 (5\theta^2 + 2\beta_1/\theta) + \nu_1 (-2\theta^3 - 4\beta_1) + (2\theta\beta_1 + \beta_1^2/\theta^2 - 4\beta_0) = 0. \quad (29)$$

Once we find the solutions of (29) we can choose a suitable one and find ν_0 , and then have an idea of which α_j generate our model. For practical reasons, since $\beta_0 = \lambda_1 \lambda_2 \lambda_3 \lambda_4$, by using (24) we can find α_1 by posing $\alpha_0 = \alpha_2 = 1$ in (24):

$$\alpha_1 = \nu_1 = \frac{\nu_0\theta^2 + \nu_0^2 - \beta_0}{\nu_0\theta}. \quad (30)$$

In our study on the model parameters, we see that is convenient to have one solution close to zero (which guarantees memory to the system). We can choose $\lambda_1 = 1/a$ (where a is a parameters which represent the memory of the system, since the saturation time is proportional to a). Since the modulus of the other solutions determine the shape of the Impulse response, is convenient to write the solutions as:

$$\begin{aligned}\lambda_1 &= c_1 \theta = 1/a \\ \lambda_2 &= c_2 \theta \\ \lambda_3 &= c_3 \theta \\ \lambda_4 &= c_4 \theta\end{aligned}\tag{31}$$

where $\sum c_j = 2$ and c_2, c_3, c_4 as to be similar among them (but not too much to avoid numerical error) to give a quick Impulsive Response.

Now we are allowed to choose suitable solutions w.r.t. the model and then find the value of γ/μ which guarantee convergence and the best fitting performance, both for first and second order.

7.3 From continuos to discrete model

In the practical implementation is not convenient to use the continuos updating formulas (4),(8), since we have to store too much value of the gradient, that is, the wider is memory of the system, the bigger is the number of elements that we have to remember. For this reason, is convenient to use a discretization of the system. Indeed, when we have a linear differential equation of order bigger than one, we can transform it in a system of the same order with only linear differential equations of order one. In our case in

$$D^4 y + \beta_3 D^3 y + \beta_2 D^2 y + \beta_1 D y + \beta_0 y + \eta \sum_{k=1}^l \zeta_k \cdot \delta(t - t_k) = 0$$

we can substitute $y_0 = y, y_1 = Dy, \dots$ and posing $u(t) = \eta \sum_{k=1}^l \zeta_k \cdot \delta(t - t_k)$ so that to have:

$$\begin{cases} y_1 &= y'_0 \\ y_2 &= y'_1 \\ y_3 &= y'_2 \\ y_4 &= -\beta_3 y_3 - \beta_2 y_2 - \beta_1 y_1 - \beta_0 y_0 - u(t) \end{cases}\tag{32}$$

and then we have the system

$$\dot{\mathbf{y}} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\beta_0 & -\beta_1 & -\beta_2 & -\beta_3 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} + \mathbf{B}u = \mathbf{A}\mathbf{y} + \mathbf{B}u \quad (33)$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\beta_0 & -\beta_1 & -\beta_2 & -\beta_3 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

from the Lagrange formula we have

$$\mathbf{y}(t) = e^{\mathbf{A}(t-t_0)}\mathbf{y}(t_0) + \int_{t_0}^t e^{\mathbf{A}(t-s)} \cdot \mathbf{B}u(s)ds. \quad (34)$$

When we consider an equally spaced discretization of the time of width τ , a general instant of time is $t = \tau K$ and if we assume $t_0 = 0$, the general evolution of the system can be computed by

$$\mathbf{y}[K] = \mathbf{y}(\tau K) = e^{\mathbf{A}\tau K}\mathbf{y}[0] + \int_0^{\tau K} e^{\mathbf{A}(\tau K-s)} \cdot \mathbf{B}u(s)ds \quad (35)$$

and at the next step we have

$$\begin{aligned} \mathbf{y}[K+1] &= e^{\mathbf{A}\tau(K+1)}\mathbf{y}[0] + \int_0^{\tau(K+1)} e^{\mathbf{A}(\tau(K+1)-s)} \cdot \mathbf{B}u(s)ds \\ &= e^{\mathbf{A}\tau} \left(e^{\mathbf{A}\tau K}\mathbf{y}[0] + \int_0^{\tau K} e^{\mathbf{A}(\tau K-s)} \cdot \mathbf{B}u(s)ds \right) + \\ &\quad + \int_{\tau K}^{\tau(K+1)} e^{\mathbf{A}[\tau(K+1)-s]} \cdot \mathbf{B}u(s)ds \\ &= e^{\mathbf{A}\tau}\mathbf{y}[K] + \int_{\tau K}^{\tau(K+1)} e^{\mathbf{A}[\tau(K+1)-s]} \cdot \mathbf{B}u(s)ds \end{aligned}$$

since $u(t)$ is composed by a summation of impulses we have a summation of integrals in the second term. If we assume that each impulse (i.e. each

supervision) is provided in the middle of two step, we have that only the integrals referred to the last impulse (the one provided at $h = \tau K + \tau/2$) is different from 0 :

$$\begin{aligned}
\int_{\tau K}^{\tau(K+1)} e^{\mathbf{A}(\tau(K+1)-s)} \cdot \mathbf{B} u(s) ds &= \eta \sum_{k=1}^l \int_{\tau K}^{\tau(K+1)} e^{\mathbf{A}[\tau(K+1)-s]} \cdot \mathbf{B} \zeta_k \cdot \delta(s - t_k) \\
&= \eta \int_{\tau K}^{\tau(K+1)} e^{\mathbf{A}[\tau(K+1)-s]} \cdot \mathbf{B} \zeta_h \cdot \delta[s - (\tau K + \tau/2)] \\
&= \eta e^{\mathbf{A}[\tau(K+1) - (\tau K + \tau/2)]} \cdot \mathbf{B} \zeta_h \\
&= \eta e^{\mathbf{A}\tau/2} \cdot \mathbf{B} \zeta_h
\end{aligned}$$

that is

$$\mathbf{y}[K+1] = e^{\mathbf{A}\tau} \mathbf{y}[K] + e^{\mathbf{A}\tau/2} \cdot \mathbf{B} \eta \zeta_h. \quad (36)$$

References

- [1] Salvatore Frandina, Marco Gori, Marco Lippi, Marco Maggini, and Stefano Melacci. Variational foundations of online backpropagation. In *Artificial Neural Networks and Machine Learning - ICANN 2013 - 23rd International Conference on Artificial Neural Networks, Sofia, Bulgaria, September 10-13, 2013. Proceedings*, pages 82–89, 2013.
- [2] Alessandro Betti and Marco Gori. The principle of least cognitive action. *Theoretical Computer Science*, pages 83 – 99, 2015.
- [3] Marco Gori, Marco Maggini, and Alessandro Rossi. Neural network training as a dissipative process. *Neural Networks*, 81:72 – 80, 2016.
- [4] Marco Maggini and Alessandro Rossi. *On-line Learning on Temporal Manifolds*, pages 321–333. Springer International Publishing, Cham, 2016.